

## 1. Code

First, `tri_dup_mode` is a Boolean that is used to control which retransmission mode to use. By setting it to True, fast retransmission with 3 duplicate acks is chosen. Setting it to false will make the program rely only on timeouts.

`lost_pkt_td` is a list to store retransmitted packet numbers when in fast retransmit mode. `lost_pkt_to` is a list to store retransmitted packet numbers when in timeout only mode. The length of list is printed at the end of program to show results. `latency_sum` is a global variable used to keep track of latency throughout the program. This variable is used with `no_pkt` at the end of program to compute latency and throughput.

The `prev_ack` variable inside the thread target function is used to check whether the received ack is a duplicate ack. If it is, global variable `dup_count` is incremented which is a variable to keep track of number of duplicate acks. When `dup_count` exceeds 3, the program will print '3 dup acks detected', appends packet number to list and retransmits using `tdupack_flag`. `latency_sum` is updated on every reception.

In the main function, a condition is added for retransmitting when 3 duplicate acks are detected. This branch will only execute when `tri_dup_mode` is True.

On either timeout or 3 duplicate acks, `ssthresh` and `win` variables are updated. On 3 duplicate acks, `win` is set to half of win (`win//2`) and `ssthresh` is set to half of win as well. On timeout, `win` is set to 1 and `ssthresh` is updated to half of win. A guard condition is set to prevent `ssthresh` into being set to 0.

## 2. 3 Duplicate Acks vs Timeout

Below is a comparison of 3 duplicate acks and timeout with 1000 total packets, and loss rates of 1%, 5%, 10%, 15% and 20%.

|  |   |   |   |  |
|--|---|---|---|--|
| fast retransmit mode(3 dup acks)<br>3 dup loss: 9<br>loss rate: 0.01<br>-----<br>latency: 0.0008851346969604492<br>throughput: 1129.771551645188<br>done | fast retransmit mode(3 dup acks)<br>3 dup loss: 49<br>loss rate: 0.05<br>-----<br>latency: 0.004865984439849854<br>throughput: 285.50826094110081<br>done | fast retransmit mode(3 dup acks)<br>3 dup loss: 96<br>loss rate: 0.1<br>-----<br>latency: 0.00890127285848694<br>throughput: 112.34349353995395<br>done | fast retransmit mode(3 dup acks)<br>3 dup loss: 170<br>loss rate: 0.15<br>-----<br>latency: 0.018914678573608398<br>throughput: 52.86899251860919<br>done | fast retransmit mode(3 dup acks)<br>3 dup loss: 199<br>loss rate: 0.2<br>-----<br>latency: 0.024016147613525392<br>throughput: 41.63865146451802<br>done |
|--|---|---|---|--|

Comparison of 3 duplicate ack latency & throughput

|   |  |   |   |  |
|---|--|---|---|--|
| timeout mode<br>timeout loss: 15<br>loss rate: 0.01<br>-----<br>latency: 0.01237834988483204<br>throughput: 88.83845722833848<br>done | timeout mode<br>timeout loss: 44<br>loss rate: 0.05<br>-----<br>latency: 0.03042789936865796<br>throughput: 25.362751153764314<br>done | timeout mode<br>timeout loss: 96<br>loss rate: 0.1<br>-----<br>latency: 0.08066131925582885<br>throughput: 12.397516848905146<br>done | timeout mode<br>timeout loss: 135<br>loss rate: 0.15<br>-----<br>latency: 0.09886782169342041<br>throughput: 10.114514337148992<br>done | timeout mode<br>timeout loss: 211<br>loss rate: 0.2<br>-----<br>latency: 0.1511297194957733<br>throughput: 6.616832237473764<br>done |
|---|--|---|---|--|

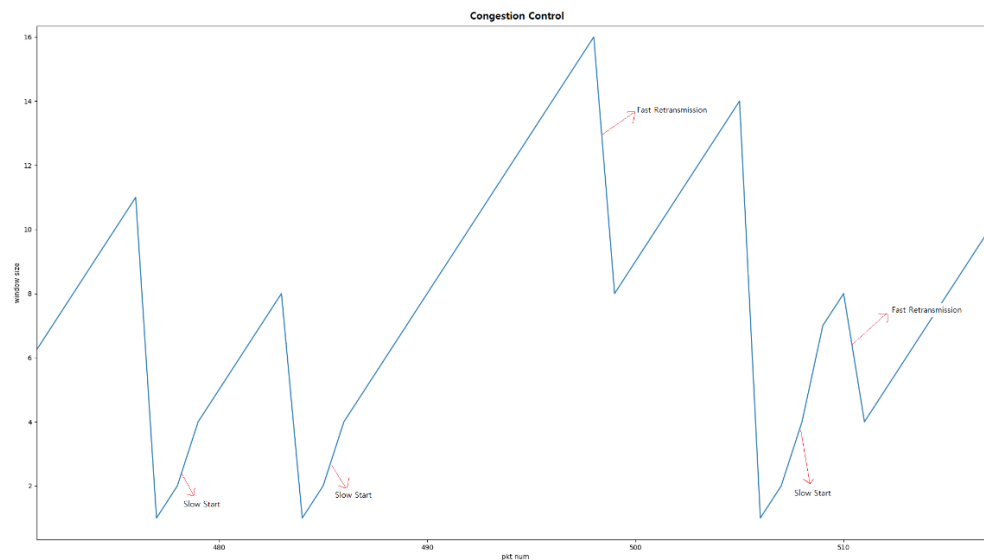
Comparison of timeout latency & throughput

It can clearly be seen that fast retransmission has much higher throughput than using timeout only. Since it is clear that a packet is lost when multiple duplicate acks are being received, it is reasonable to retransmit when 3 duplicate acks are detected. Throughput

decreases dramatically from loss rate 1% to 5%. We can also observe that throughput and loss rate is inversely proportional.

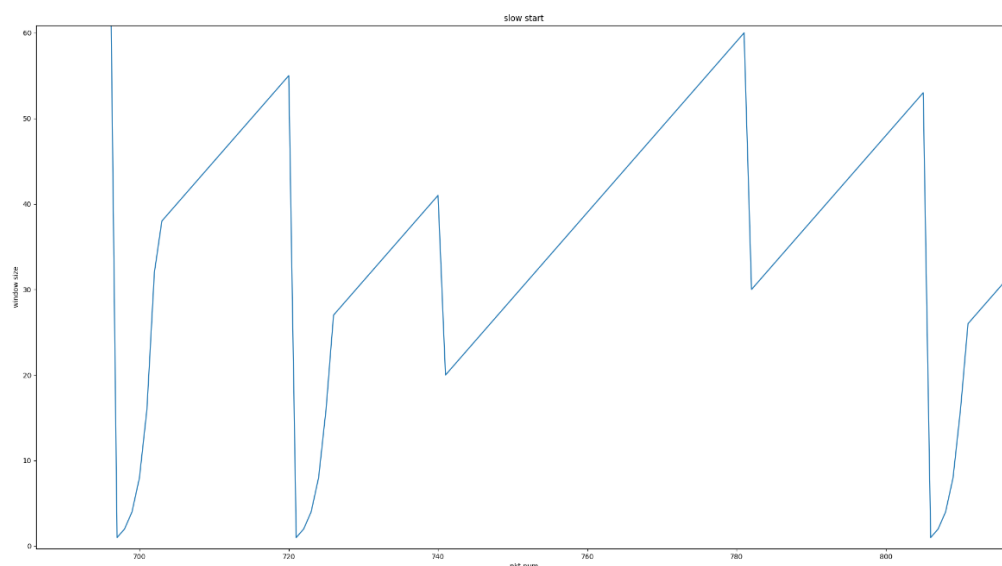
### 3. Slow Start

Below is a sample execution of slow start, plotted with matplotlib. Number of packets was 1000, loss rate was 10%, server queue size was 20 and queueing delay was 0.2.



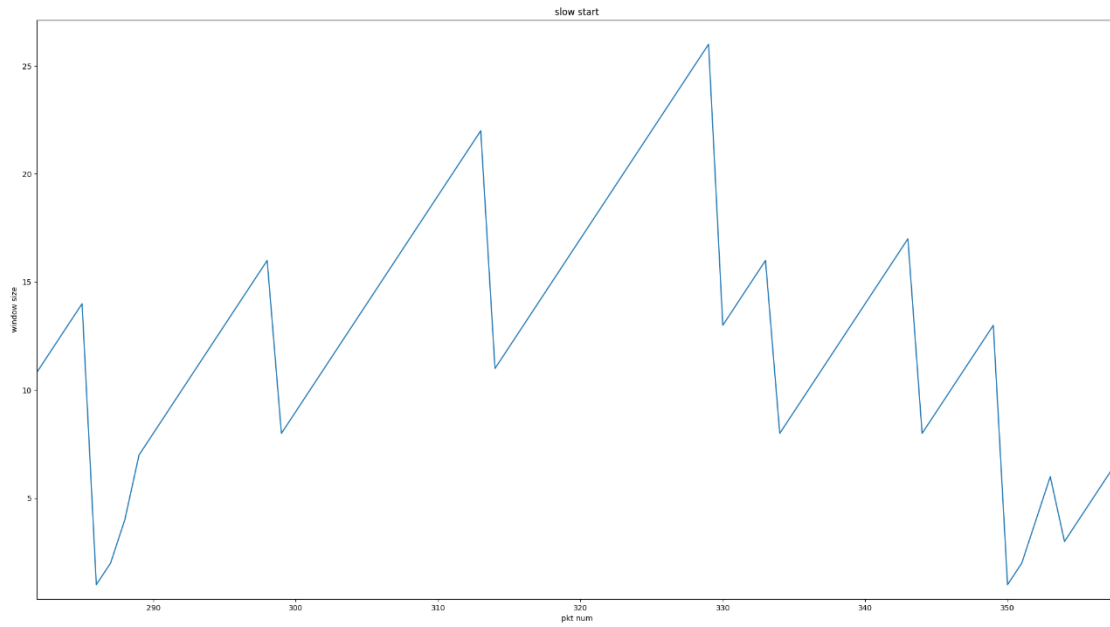
```
loss rate: 0.1
-----
latency: 0.04077153706550598
throughput: 24.526914410740524
```

Below is another sample execution with same number of packets, server queue size and queueing delay, and loss rate was set to 5%.



```
loss rate: 0.05
-----
latency: 0.032848089456558226
throughput: 30.44317086759354
```

Sample executions with queueing delay of 0.4 was also tested, but execution time was noticeably high and was clear that performance is worse compared to the sample executions above.



```
loss rate: 0.05
```

```
-----  
latency: 0.14466022396087647  
throughput: 6.912750254489107
```